Transport Layer

Gursharan Singh Tatla
mailme@gursharansingh.in
www.eazynotes.com

Introduction

- The transport layer is the fourth layer from the bottom in the OSI reference model.
- It is responsible for message delivery from process running in source computer to the process running in the destination computer.
- Transport layer does not perform any function in the intermediate nodes.
- It is active only in the end systems.

Introduction

- Data Link Layer is responsible for delivery of frames between two neighboring nodes over a link.
  - This is called ***node-to-node delivery***.
- Network Layer is responsible for delivery of datagrams between two hosts.
  - This is called ***host-to-host delivery***.
- Transport Layer is responsible for delivery of entire message from one process running on source to another process running on destination.
  - This is called ***process-to process delivery***.

Transport Layer Design Issues

- The transport layer delivers the message from one process to another process running on two different hosts.
- Thus, it has to perform number of functions to ensure the accurate delivery of message.
- The various functions of transport layer are:
  - Establishing, Maintaining & Releasing Connection
  - Addressing
  - Data Transfer
  - Flow Control
  - Error Control
  - Congestion Control

Transport Layer Design Issues

- **Establishing, Maintaining & Releasing Connection:**
  - The transport layer establishes, maintains & releases end-to-end transport connection on the request of upper layers.
  - Establishing a connection involves allocation of buffers for storing user data, synchronizing the sequence numbers of packets etc.
  - A connection is released at the request of upper layer.

Transport Layer Design Issues

- **Addressing:**
  - In order to deliver the message from one process to another, an addressing scheme is required.
  - Several process may be running on a system at a time.
  - In order to identify the correct process out of the various running processes, transport layer uses an addressing scheme called ***por number***.
  - Each process has a specific port number.

Transport Layer Design Issues

- **Data Transfer:**
  - Transport layer breaks user data into smaller units and attaches a transport layer header to each unit forming a TPDU (TransPort Layer Data Unit).
  - The TPDU is handed over to the network layer for its delivery to destination.
  - The TPDU header contains port number, sequence number, acknowledgement number, checksum and other fields.

Transport Layer Design Issues

- **Flow Control:**
  - Like data link layer, transport layer also performs flow control.
  - However, flow control at transport layer is performed end-to-end rather than node-to-node.
  - Transport Layer uses a sliding window protocol to perform flow control.

Transport Layer Design Issues

- **Error Control:**
  - Transport layer also provides end-to-end error control facility.
  - Transport layer deals with several different types of errors:
    - Error due to damaged bits.
    - Error due to non delivery of TPDUs.
    - Error due to duplicate delivery of TPDUs.
    - Error due to delivery of TPDU to a wrong destination.

Transport Layer Design Issues

- **Congestion Control:**
  - Transport layer also handles congestion in the networks.
  - Several different congestion control algorithms are used to avoid congestion.

Transport Layer Services

- Transport layer protocols can provide two types of services:
  - Connection Oriented Service
  - Connectionless Service

Transport Layer Services

- **Connection Oriented Service:**
  - In connection oriented service, a connection is first established between sender and the receiver.
  - Then, transfer of user data takes place.
  - At the end, connection is released.
  - The connection oriented service is generally reliable.
  - Transport layer protocols that provide connection oriented service are **TCP** and **SCTP** (Stream Control Transmission Protocol).

Transport Layer Services

- **Connectionless Service:**
  - In the service, the packets are sent from sender to receiver without the establishment of connection.
  - In such service, packets are not numbered.
  - The packets may be lost, corrupted, delayed or disordered.
  - Connectionless service is unreliable.
  - Transport layer protocol that provides this service is **UDP**.

Elements of Transport Protocols

- **Addressing:**
  - In order to deliver data from one process to another, address is required.
  - In order to deliver data from one node to another, MAC address is required.
  - Such an address is implemented at Data Link Layer and is called **Physical Addressing**.

Elements of Transport Protocols

- **Addressing (Cont.):**
  - In order to deliver data from one network to another, IP address is required.
  - Such an address is implemented at Network Layer and is called **Logical Addressing**.
  - Similarly, in order to deliver data from a process running on source to process running on destination, transport layer defines the **Service Point Address** or **Port Numbers**.

Elements of Transport Protocols

- **Port Numbers:**
  - Each communicating process is assigned a specific port number.
  - In order to select among multiple processes running on a destination host, a port number is required.
  - The port numbers are 16-bit integers between 0 and 65,535.

Elements of Transport Protocols

- **Port Numbers (Cont.):**
  - Port numbers are assigned by Internet Assigned Number Authority (IANA).
  - IANA has divided the port numbers in three categories:
    - **Well Known Ports:** The ports ranging from 0 to 1023. For e.g.: HTTP: 80, SMTP: 25, FTP: 21.
    - **Registered Ports:** The ports ranging from 1024 to 49,151. These are not controlled by IANA.
    - **Dynamic Ports:** The ports ranging from 49,152 to 65,535. These can be used by any process.

Elements of Transport Protocols

- **Socket Address:**
  - Socket address is a combination of IP address and port number.
  - In order to provide communication between two different processes on different networks, both IP address and port number, i.e. socket address is required.

Elements of Transport Protocols

- **Multiplexing & Demultiplexing:**
  - A network connection can be shared by various applications running on a system.
  - There may be several running processes that want to send data and only one transport layer connection available, then transport layer protocols may perform multiplexing.
  - The protocol accepts the messages from different processes having their respective port numbers, and add headers to them.

Elements of Transport Protocols

- **Multiplexing & Demultiplexing (Cont.):**
  - The transport layer at the receiver end performs demultiplexing to separate the messages for different processes.
  - After checking for errors, the headers of messages are dropped and each message is handed over to the respective processes based on their port numbers.

Elements of Transport Protocols

- **Connection Establishment:**
  - Before communicating, the source device must first determine the availability of the other to exchange data.
  - Path must be found through the network by which the data can be sent.
  - This is called Connection Establishment.

Elements of Transport Protocols

- **Connection Establishment (Cont.):**
  - Connection establishment involves **Three-Way Handshaking** mechanism:
    - The source sends a **connection request** packet to the destination.
    - The destination returns a confirmation packet back to the source.
    - The source returns a packet acknowledging the confirmation.

Elements of Transport Protocols

- **Connection Release:**
  - Once all of the data has been transferred, the connection must be released.
  - It also requires a **Three-Way Handshaking** mechanism:
    - The source sends a **disconnect request** packet to the destination.
    - The destination returns a confirmation packet back to the source.
    - The source returns a packet acknowledging the confirmation.

Transport Layer Protocols

- Transport layer provides two types of services:
  - Connection Oriented Service
  - Connectionless Service
- For this, transport layer defines two different protocols:
  - Transmission Control Protocol (TCP)
  - User Datagram Protocol (UDP)

Transmission Control Protocol

- Transmission Control Protocol (TCP) is a connection oriented protocol that provides reliable services between processes on different hosts.
- It uses the services of lower layer which provide connectionless and unreliable service.

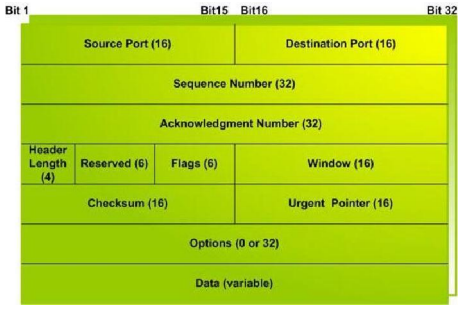
Transmission Control Protocol

- The basic features of TCP are:
  - It provides efficient method for numbering different bytes of data.
  - It provides stream data transfer.
  - It offers reliability.
  - It provides efficient flow control.
  - It provides full duplex operation.
  - It provides multiplexing.
  - It provides connection oriented service.

TCP Segment

- TCP segment is the unit of data transferred between two processes.
- Each TCP segment consists of two parts:
  - Header Part
  - Data Part

Format of TCP Segment



Format of TCP Segment

- **Source Port:**
  - It indicates the port number of a source process. It is of 2 bytes.
- **Destination Port:**
  - It indicates the port number of destination process. It is also 2 bytes.
- **Sequence Number:**
  - It specifies the number assigned to the current message. It is of 4 bytes.

Format of TCP Segment

- **Acknowledgement Number:**
  - It indicates the sequence number of the next byte of data. It is of 4 bytes.
- **Header Length:**
  - It indicates number of words in the TCP header. It is a 4 bit field.
- **Reserved:**
  - This 6 bit field is reserved for future use.

Format of TCP Segment

- **Flags:**
  - This 6 bit field consists of 6 different flags:
    - UGR (Urgent Pointer)
    - ACK (Acknowledgement)
    - PSH (Request for Push)
    - RST (Reset the Connection)
    - SYN (Synchronize)
    - FIN (Final or Terminate the Connection)

Format of TCP Segment

- **Window:**
  - It specifies the size of sender's receiving window, i.e., the buffer space available for incoming data. It is of 2 bytes.
- **Checksum:**
  - This 16-bit field contains the checksum.
- **Urgent Pointer:**
  - This 16-bit field is valid only if urgent pointer in flags is set to 1.

Format of TCP Segment

- **Options:**
  - It contains the optional information in the TCP header. It is of 32 bytes.
- **Data:**
  - This field contains the upper layer information. It is of variable size.

User Datagram Protocol

- User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol.
- Like TCP, UDP also provides process-to-process communication.
- Unlike TCP, it does not provide flow control and error control mechanisms.
- It is connectionless, therefore, it transfers data without establishing a connection.

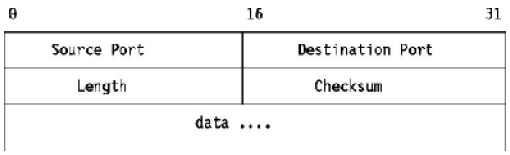
User Datagram Protocol

- The various features of UDP are:
  - It provides connectionless transport service.
  - It is unreliable.
  - It does not provide flow control and error control.
  - It is less complex and is simple than TCP, and easy to implement.
  - User datagrams (packets) are not numbered.

UDP Datagram

- A datagram is the unit of data transferred between two processes.
- Each UDP datagram consists of two parts:
  - Header Part
  - Data Part.

Format of UDP Datagram



UDP Datagram

- **Source Port:**
  - It indicates the port number of source process. It is of 16 bits.
- **Destination Port:**
  - This 16 bit field specifies the port number of destination process.

UDP Datagram

- **Length:**
  - It specifies the total length of the user datagram (header + data). It is of 16 bits.
- **Checksum:**
  - The contains the checksum, and is optional. It is also of 16 bits.

Thank You

Have a Nice Day